REPORT DOCUMENTATION PAGE

Form Approved OMB NO. 0704-0188

1. REPORT DATE (DD-MM-YYYY)	2. REPORT TYPE	3. DATES COVERED (From - To)
	Technical Report	-

4. TITLE AND SUBTITLE

A Mathematical and Sociological Analysis of Google Search Algorithm

5a. CONTRACT NUMBER
W911NF-11-1-0322

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
611102

6. AUTHORS

Ming-Jun Lai, Dawn Robinson

5d. PROJECT NUMBER

5e. TASK NUMBER

5f. WORK UNIT NUMBER

7. PERFORMING ORGANIZATION NAMES AND ADDRESSES

University of Georgia Research Foundation, Inc.
Sponsored Research Office
University of Georgia Research Foundation Inc
Athens, GA 30602 -

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U.S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSOR/MONITOR'S ACRONYM(S)
ARO

11. SPONSOR/MONITOR'S REPORT NUMBER(S)
60673-MA.2

12. DISTRIBUTION AVAILIBILITY STATEMENT

Approved for public release; distribution is unlimited.

13. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not contrued as an official Department of the Army position, policy or decision, unless so designated by other documentation.

14. ABSTRACT

Google search algorithm for finding relevant information on-line based on keywords, phrases, links, and webpages is analyzed in the mathematical and sociological settings in this article.
We shall first survey mathematical study related to the Google search engine and then present a new analysis for the convergence of the search algorithm and a new update scheme.

15. SUBJECT TERMS

Google Matrix, PageRank, In-and Out-Linkages, Secondary Linkages

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	15. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON
a. REPORT	b. ABSTRACT	c. THIS PAGE			Ming-Jun Lai
UU	UU	UU	UU		19b. TELEPHONE NUMBER 706-542-2065

Report Title

A Mathematical and Sociological Analysis of Google Search Algorithm

ABSTRACT

Google search algorithm for finding relevant information on-line based on keywords, phrases, links, and webpages is analyzed in the mathematical and sociological settings in this article.

We shall first survey mathematical study related to the Google search engine and then present a new analysis for the convergence of the search algorithm and a new update scheme.

Next based on sociological knowledge, we propose to use in- and out- linkages as well as use the second order linkages to refine and improve the search algorithm. We use the sociology to justify our proposed improvements and mathematically prove the convergence of these two new search algorithms.

A Mathematical and Sociological Analysis of Google Search Algorithm *

Ming-Jun Lai† Dawn Robinson‡

January 16, 2013

Abstract

Google search algorithm for finding relevant information on-line based on keywords, phrases, links, and webpages is analyzed in the mathematical and sociological settings in this article. We shall first survey mathematical study related to the Google search engine and then present a new analysis for the convergence of the search algorithm and a new update scheme. Next based on sociological knowledge, we propose to use in- and out- linkages as well as use the second order linkages to refine and improve the search algorithm. We use the sociology to justify our proposed improvements and mathematically prove the convergence of these two new search algorithms.

1 Introduction

Due the huge volume of documents available through the world wide web, a search engine or similar service is now absolutely necessary in order to find relevant information on-line. About 12 years ago, Google, Yahoo and other companies started providing such a service. It is interesting to know the mathematics of computational algorithms behind these search engines, in particular, the Google search engine which is so successful nowaday. Many mathematicians and computer scientists, G. Golub and his collaborators, Brezinski and his collaborators, Langville and Meyer, among other pioneers worked on the mathematics of the Google matrix and its computation. See [25], [16], [9], [8], [31], [32] and a recent survey in [2]. The search algorithm also becomes a very useful tool in many Web search technologies such as spam detection, crawler configuration, trust networks, and etc.. and find many applications in [12], [13] and etc.. Today, it is the most used computational algorithm in the world. More and more websites such as Facebook, Amazon, Netflix, and etc.. use similar search algorithms for various purposes.

As almost all people on the earth use the Google search once or several times daily, it is interesting to see if the search results make any sense in sociology. How could it be adjusted to be more effective sociologically. We divide this article into two parts. We shall first present a mathematical description of the computational algorithm behind the Google search engine and summarize the recent studies related to the algorithm. Then we present a sociological analysis. Based on the sociology, we propose two modifications in order to be more effective.

*This research is supported by Army Research Office Grant # W911NF–11–1–0322.

†mjlai@math.uga.edu. Department of Mathematics, The University of Georgia, Athens, GA 30602.

‡sodawn@uga.edu. Department of Sociology, The University of Georgia, Athens, GA 30602.

2 Mathematical Analysis

In this section we first survey the mathematical studies related to the Google search algorithm. Then we present a mathematical analysis which is new to the best of our knowledge. Mainly we study its convergence as it is an iterative method since the Google matrix is of huge size so that any direct method is impossible. Then we shall establish the convergence, present the rate of convergence and a local updating scheme.

2.1 Google Search Engine

During each search, after entering a key word or key words, the Google search engine will produce a ranking vector called PageRank. It is a vector of huge size (several billion entries) although almost all entries are zero in general. Each entry of the vector presents a webpage. Nonzero value in an entry is a weighting factor deciding how important the page related to the key word(s) just entered. The importance is determined by popularity. The ideas for this computational algorithm have been reported as early as 1997–1998 by Page in [37] and Brin and Page in [10]. See also [33] for explanation of the ideas. The algorithm mainly computes a numerical weighting (called pagerank) to each webpage or each element in a hyperlinked set of documents. Pageranks reflect the Google's view of the importance of webages among more than several billion webpages. It is a pragmatic approach to improve search quality by going through the collective intelligence of the web to determine a page's importance.

Let $\mathbf{v}$ be a vector of $\mathbb{R}^N$ with $N \geq 8\ billion$. Any unit vector in $\mathbb{R}^N$ is associated with a webpage. Mathematically, Pagerank is a probability function of vectors. For a page $\mathbf{v}$, let $P(\mathbf{v})$ be the PageRank of $\mathbf{v}$. By a stroke of the ingenious, Brin and Page thought that $P(\mathbf{v})$ is the weighted sum of the PageRanks of all pages pointing to $\mathbf{v}$. That is,

$$P(\mathbf{v}) = \sum_{\mathbf{u} \in A_{\mathbf{v}}} P(\mathbf{u}) \ell(\mathbf{u}, \mathbf{v}), \tag{1}$$

where $A_{\mathbf{v}}$ is the collection of all pages connected to $\mathbf{v}$ and $\ell(\mathbf{u}, \mathbf{v}) \in [0, 1]$ be a weight dependent on the number of links from page $\mathbf{u}$ to the other pages. More precisely, let $L(\mathbf{u})$ be the number of linkages from $\mathbf{u}$ and then define $\ell(\mathbf{u}, \mathbf{v}) = 1/L(\mathbf{u})$ if $\mathbf{u} \in A_{\mathbf{v}}$ and 0 otherwise for any other page $\mathbf{u}$. Such a definition is cyclic. Also, a web surfer may get bored and abandons the current search by teleporting to a new page. By a second stroke of ingenious, Brin and Page introduced a damping factor and modified the Pagerank function $P(\mathbf{v})$ to be

$$P(\mathbf{v}) = d \sum_{\mathbf{u} \in A_{\mathbf{v}}} P(\mathbf{u}) \ell(\mathbf{u}, \mathbf{v}) + \frac{1-d}{N}, \quad \forall \text{unit vector } \mathbf{v} \in \mathbb{R}^N, \tag{2}$$

where $d \in (0, 1)$ is a damping factor. It indeed leads to a convergent algorithm and the PageRank P will be unique defined. In summary, we have

- PageRank
$$\mathbf{P} = [P(\mathbf{v}_1), \cdots, P(\mathbf{v}_N)]^\top. \tag{3}$$
- $M = [\ell(\mathbf{v}_i, \mathbf{v}_j)]_{1 \leq i,j \leq N}$ be the adjacency matrix of size $N \times N$ with convention $\ell(\mathbf{v}_i, \mathbf{v}_i) = 0$.
- $\mathbf{1} = [1, 1, \cdots, 1]^\top \in \mathbb{R}^N$;

- The Google Matrix

$$\mathcal{G} = dM + \frac{1-d}{N}\mathbf{1}\cdot\mathbf{1}^\top. \tag{4}$$

which is a stochastic matrix, where $d > 0$ is a damping factor. In Google search engine, $d = 0.85$.

To compute the PageRank $\mathbf{P}$, the Google search engine computes: starting with an inital vector $\mathbf{P}_1, e.g., \mathbf{P}_1 = \mathbf{1}/N$,

$$\mathbf{P}_{k+1} = \mathcal{G}\mathbf{P}_k, \forall k = 1, 2, \cdots \tag{5}$$

until the difference of $\mathbf{P}_{k+1}$ and $\mathbf{P}_k$ within a given tolerance.

For an elementary explanation of Google computational algorithm, see a monograph [33]. Many mathematical problems arise from the algorithm. For example, does the computation (5) converge and what is the convergence rate? Is the computation stable? How to speed up the iterations in (5)? How to do the computation in parallel? Can we have $d = 1$? How can we add or delete a page web from the PageRank function?

2.2 Summary of Mathematical Results

The major question for the computational algorithm (5) in the previous subsection is the convergence rate of the iterations in (5). The iteration is a power method. The convergence is dependent on the second largest eigenvalue as the first largest eigenvalue is 1 (cf. [27]). The value of the second largest eigenvalue or eigenvalues is the damping factor as shown first in [24]. This fact is later established in [39] by using Jordan canonical form of the Google matrix. It was also proved in [14] where the researcher showed that the second largest eigenvalue is the dampling factor even the Google matrix has more than 1 largest eigenvalues. It is also proved in [30]. In [11], the searcher interpreted the PageRank algorithm as a power series which leads to a new proof of the several known results about the PageRank, e.g.., the convergence rate is the exactly the damping factor by [24] and presented a new inside about PageRank which is the probability that a specific type of random surfer ends his walk at the page. Usually, the random surfer is assumed to be uniform distribution suggested by Page and Brin. In fact, all the properties of the PageRank hold for arbitrary probability setting. We shall present another proof in the next subsection

A second question is if the computation is stable or what the condition number of the Google matrix is. In [18], the researchers viewed the PageRank vector to the stationary distribution of a stochastic matrix, the Google matrix. They analyzed the sensitivity of PageRank to changes in the Google matrix, including addition and deletion of links in the web graph and presented error bounds for the iterates of the power method and for their residuals. See also various properties in [9] including the condition number of the Google matrix.

A central question is how to accelerate the Google computational algorithm. There are several approaches.

- *Extrapolation* In [8], the researchers studied the computation of the nonnegative left eigenvector associated with the dominant eigevalue of the PageRank matrix and proposed to use extrapolation methods to approximate the eigenvector. Furthermore, in [9], the researchers proposed some acceleration methods due to the slow convergence of the Power method for the PageRank vector. The acceleration metod can be explained as the method of Vorobyev moments and Krylov subspace method. In addition, many properties about the Google matrix and PageRank vectors were presented.

- *Sparsity Split* In [40], these Chinese researchers use the sparsity of the hyperlink matrix (one of the two matrices in Google Matrix) and proposed a linear extrapolation to optimize each iteration in the Power method for the computation of PageRank and reduction of the storage space. The idea is simple and the engineers in Google may have already implemented.

- *Iterative Solutions* The iteration in (5) can be reinterpreted as a linear system to be given in (8). As explained in [15], the Google matrix is very large, sparse and non-symmetric. Solution of the linear system by a direct method is not feasible due to the matrix size. Many iterative solutions such as Gauss-Jacobi iterations, Generalize Minimum Residual (GMRES), Biconjugate Gradient (BiCG), Quasi-Minimal Residual (QMR), Conjugate Gradient Squared (CGS), Biconjugate Gradient Stabilized (BiCGSTAB), Chebyshev Iterations. The researchers proposed to use the parallel Block Jabobi iteration with adaptive Schwarz preconditioners.

- *Parallel Iteration* In [15], the researchers described a parallel implementation for PageRank computation and demonstrated that the parallel PageRank computing is faster and less sensitive to the changes in teleportation.

- *Lumping Dangling Nodes* In [19], the researchers lumped all of the dangling nodes together into a single node and the Google matrix can be reduced. They showed that the reduced stochastic matrix has the same nonzero eigenvalues as the full Google matrix and the convergence rate is the same when the Power method is applied. In [34], the researchers further reduced the Google matrix by lumping more nodes which are so-called weakly nondangling nodes together and the reduced matrix has the same nonzero eigenvalues as the Google matrix.

- *Distributed Randomized Algorithm* In a series of papers ([20], [21], [22], the researchers applied distributed randomized algorithm for PageRank. The idea is to let each page compute its own value by exchanging information with its linked pages. The communication among the pages is randomized so as to make it asynchronous. The communication among pages in the sense that data transmitted over the links is received correctly or could have some noises.

There are several other improvements. For example,

One is to update the PageRank. In [33], the researchers considered the PageRank vector to be an state of homogeneous irreducible Markov chain and the chain requires updating by altering some of its transition probabilities or by adding or deleting some states. They proposed a general purpose algorithm which simultaneously deals with both kinds of updating problems and proved its convergence. We shall present another approach for local updating to be discussed in the next subsection.

Another improvement is to distinct the webpage with equal rankings. In [35], the researchers proposed an idea to organize the search result produced by the PageRank where several pages have the same rank score so that the surfer can get more relevant and important results easily. Their idea is to add a weight to each page.

We finally remark that besides the PageRank algorithm, there are many other search algorithms available. For example, Kleinberg's HITS algorithm is a classic Hypertext Induced Topics Search (HITS) algorithm invented by Kleinberg (cf. [28]). Kleinberg viewed that most pages are a hub and authority, i.e., a general page not only points to other pages and also receives points from other pages. Again, the HITS algorithm uses the power method to the hub weight vector and authority weigh vector. See [33] and

[6] for more explanation. For another example, Yahoo uses another type of algorithms. See last section of this paper.

In summary, the study of Google computational algorithm requires many mathematical tools such that graph theory (cf. [**?**]), random walks and Markov chains (c.g., [4]. [33]), numerical analysis (cf. [16]), [15], [8]), stochastic matrices (cf. [18]), networks theory (cf. [6]), etc.. See [2] for a survey of recent techniques for on link-based ranking methods and the PageRank algorithm to find the most relevant documents from the WWW.

2.3 Our Convergence Analysis

We see that $P(\mathbf{v}) \geq 0$. We now claim that $P(\mathbf{v}) \leq 1$. In fact, we can prove the following

Theorem 2.1 *Let $\mathbf{v}_i, i = 1, \cdots, N$ be the standard unit vectors in $\mathbb{R}^N$. Then the PageRanks $P(\mathbf{v}_i), i = 1, \cdots, N$ satisfy*

$$\sum_{i=1}^{N} P(\mathbf{v}_i) = 1.$$

Proof. Indeed, as in the definition (2), $P(\mathbf{v}_i), i = 1, \cdots, N$ are dependent on each other. We recall

$$\mathbf{P} = [P(\mathbf{v}_1), \cdots, P(\mathbf{v}_N)]^\top$$

is the PageRank vector in $\mathbb{R}^N$. We can see that

$$\sum_{i=1}^{N} \ell(\mathbf{v}_j, \mathbf{v}_i) = 1 \tag{6}$$

for $j = 1, \cdots, N$. We call $\ell(\mathbf{v}_i, \mathbf{v}_j)$ adjacency functions. It follows from (2), we can write

$$\mathbf{P} = \frac{1-d}{N}\mathbf{1} + dM\mathbf{P}, \tag{7}$$

where $M = [\ell(\mathbf{v}_i, \mathbf{v}_j)]_{1 \leq i,j \leq N}$ be the adjacency matrix of size $N \times N$ with convention $\ell(\mathbf{v}_i, \mathbf{v}_i) = 0$ and $\mathbf{1} = [1, 1, \cdots, 1]^\top \in \mathbb{R}^N$. Hence, we have

$$\begin{aligned}\sum_{i=1}^{N} P(\mathbf{v}_i) = \mathbf{1}^\top \mathbf{P} &= N\frac{1-d}{N} + d\mathbf{1}^\top M\mathbf{P} \\ &= 1 - d + d\mathbf{1}^\top \mathbf{P} = 1 - d + d\sum_{i=1}^{N} P(\mathbf{v}_i),\end{aligned}$$

where we have used (6) to have $\mathbf{1}^\top M = \mathbf{1}^\top$. That is, we have $(1-d)\sum_{i=1}^{N} P(\mathbf{v}_i) = 1 - d$ and hence, $\sum_{i=1}^{N} P(\mathbf{v}_i) = 1$ since $d \neq 1$. This completes the proof. ■

The PageRank algorithm offers two ways to calculate the PageRank vector $\mathbf{P}$: one is by algebraic method and the other by an iterative method. Indeed, by (7), we have

$$(I - dM)\mathbf{P} = \frac{1-d}{N}\mathbf{1} \quad \text{or} \quad \mathbf{P} = \frac{1-d}{N}(I - dM)^{-1}\mathbf{1}, \tag{8}$$

where I is the identity matrix of size $N \times N$. This algebraic method will be well-defined if $I - dM$ is invertible. Indeed, we have

Theorem 2.2 *Let I be the identity matrix and M be the adjacency matrix defined above. If $d \in (0, 1)$, then*

$$I - dM$$

is invertible.

Proof. We first recall a matrix $A = [a_{ij}]_{1 \le i,j \le N}$ is diagonally dominant if

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{N} |a_{ij}|$$

for all $i = 1, \cdots, N$. It is easy to see that $(I - dM)^\top$ is diagonally dominant matrix by using (6) with $d < 1$. It is known (cf. [27], p. 178) that every diagonally dominant matrix is nonsingular, and hence is invertible. So is $I - dM$. ■

Certainly, finding the inverse $(I - dM)^{-1}$ is expensive in computation. An easy approach is to use an iterative method, for example, power method. Starting with $\mathbf{P}_0 = \mathbf{1}/N$, we iteratively compute

$$\mathbf{P}_k = \mathcal{G}\mathbf{P}_{k-1} = \frac{1-d}{N}\mathbf{1} + dM\mathbf{P}_{k-1} \tag{9}$$

by using Theorem 2.1 for $k = 1, 2, \cdots$,. We now show that $\mathbf{P}_k, k \ge 1$ converge. Let us use the ℓ_1 norm for matrices to measure the errors $\mathbf{P}_k - \mathbf{P}$. Recall the ℓ_1 norm for matrix A with $A = [a_{ij}]_{1 \le i,j \le N}$ is

$$\|A\|_1 = \max_{j=1,\cdots,N} \sum_{i=1}^{N} |a_{ij}|.$$

Then we have

Theorem 2.3 *Let $\mathbf{P}$ be the PageRank vector defined by the algebraic method. For all $k \ge 1$,*

$$\|\mathbf{P}_k - \mathbf{P}\|_1 \le Cd^k,$$

where C is a positive constant.

Proof. It is easy to see from (7) and the definition (9) of iterations that

$$\mathbf{P}_k - \mathbf{P} = dM(\mathbf{P}_{k-1} - \mathbf{P}) = \cdots = (dM)^k(\mathbf{P}_0 - \mathbf{P}).$$

Thus, letting $C = \|\mathbf{P}_0 - \mathbf{P}\|_1$,

$$\|\mathbf{P}_k - \mathbf{P}\|_1 \le d^k \|M^k\|_1 C \le Cd^k \|M\|_1^k.$$

Finally we note that $\|M\|_1 = 1$ by using (6) to complete the proof. ■

We remark that the convergence of $\mathbf{P}_k$ to $\mathbf{P}$ was shown numerically in [37] for full size and a half size link databases. The rates of convergence are nearly the same for these two sizes. Also in [23], the contributor(s) of this wiki page uses the power method for the leading eigenvector to explain the convergence of the iterative method above without giving a convergence rate as it is based on the ratio of the second largest and the largest eigenvalues of the Google matrix. The second eigenvalue of the Google matrix was studied in [24] where the rate of convergence of PageRank was determined to be 0.85, the damping factor d. As mentioned in a previous section, the convergence rate has been determined by several methods already. Our result in Theorem 2.3 gave another proof. It is much simpler and easier fashion than the study in [24].

2.4 A New Local Update Scheme

We discuss how to do local updates. That is, in practice, webpages are updated constantly in the sense that the number $L(\mathbf{v}_i)$ of links from webpage $\mathbf{v}_i$ changes very often, usually increases. A new adjacency matrix M is needed updated everyday. Instead of using an updated matrix M to compute a new PageRank vector $\mathbf{P}$ for all components, we discuss an approach to update partial components of $\mathbf{P}$. We need the following well-known formula:

Lemma 2.1 (Sherman-Morrison's formula,1949) *If A is invertible and $\mathbf{x}, \mathbf{y}$ are two vectors such that $\mathbf{y}^\top A^{-1}\mathbf{x} \neq -1$, then $A + \mathbf{x}\mathbf{y}^\top$ is invertible and*

$$(A + \mathbf{x}\mathbf{y}^\top)^{-1} = A^{-1} - \frac{A^{-1}\mathbf{x}\mathbf{y}^\top A^{-1}}{1 + \mathbf{y}^\top A^{-1}\mathbf{x}}.$$

Assume that we have an updated $\ell_+(\mathbf{v}_i, \mathbf{v}_j)$ for a fixed integer j and M_+ be the updated adjacency matrix. Let $n = n(j)$ and $n_+ = n_+(j)$ be the previous and updated numbers $L(\mathbf{v}_j)$ of links from webpage $\mathbf{v}_j$. Let us forcus the case that only more webpage links are added to the existing links from the webpage $\mathbf{v}_j$. In this case $n_+ > n$. Hence, we have

$$M_+ = M + \mathbf{x}_j \mathbf{v}_j^\top,$$

where $\mathbf{x}_j = [\ell_+(\mathbf{v}_i, \mathbf{v}_j) - \ell(\mathbf{v}_i, \mathbf{v}_j), i = 1, \cdots, N]^\top$ with $\ell_+(\mathbf{v}_i, \mathbf{v}_j)$ is the updated version of $\ell(\mathbf{v}_i, \mathbf{v}_j)$ and

$$\|M - M_+\|_1 = 1 - n/n_+ + \frac{n_+ - n}{n_+} \leq 2\frac{n_+ - n}{n_+}.$$

We shall use $\mathbf{P}_+$ to be the updated PageRank vector satisfying

$$\mathbf{P}_+ = \frac{1-d}{N}\mathbf{1} + dM_+\mathbf{P}_+.$$

By Theorem 2.2, we know $I - dM_+$ is invertible. By the inverse formula in Lemma 2.1, we know $\mathbf{v}_j^\top (I - dM)^{-1}\mathbf{x}_j \neq -1$ and hence, letting $\alpha = 1 + \mathbf{v}_j^\top (I - dM)^{-1}\mathbf{x}_j \neq 0$,

$$(1 - dM_+)^{-1} = (1 - dM)^{-1} - (I - dM)^{-1}\mathbf{x}_j \mathbf{v}_j^\top (1 - dM)^{-1}/\alpha. \tag{10}$$

It follows

$$\begin{aligned}\mathbf{P}_+ &= (1-dM_+)^{-1}\frac{1-d}{N}\mathbf{1}\\ &= (1-dM)^{-1}\frac{1-d}{N}\mathbf{1} - \frac{1}{\alpha}(I-dM)^{-1}\mathbf{x}_j\mathbf{v}_j^\top(1-dM)^{-1}\frac{1-d}{N}\mathbf{1}\\ &= \mathbf{P} - \frac{1}{\alpha}(I-dM)^{-1}\mathbf{x}_j\mathbf{v}_j^\top\mathbf{P}\\ &= \mathbf{P} - \frac{P_j}{\alpha}(I-dM)^{-1}\mathbf{x}_j. \end{aligned} \tag{11}$$

where P_j is the j^{th} component of vector $\mathbf{P}$. Let us write $\epsilon = \frac{P_j}{\alpha}(I-dM)^{-1}\mathbf{x}_j$ for the update for $\mathbf{P}$ to obtain $\mathbf{P}_+$ and we next discuss how to find the update efficiently as solving $(I-dM)^{-1}\mathbf{x}_j$ is expensive in computation.

We need the following (cf. [27], p. 198)

Lemma 2.2 (Neumann Series) *If A is an $n\times n$ matrix and $\|A\|_1 < 1$, then $I-A$ is invertible and*

$$(I-A)^{-1} = \sum_{k=0}^{\infty} A^k.$$

As shown before $\|M\|_1 = 1$ and $d\in(0,1)$, letting $A = dM$ in Lemma 2.2 above, we have

$$(I-dM)^{-1}\mathbf{x}_j = \sum_{k=0}^{\infty} d^k M^k \mathbf{x}_j.$$

Algorithm 2.1 (A Single Update) *Choose an integer K large enough such that d^K is within a given tolerance. Start with $\mathbf{y} = \mathbf{x}_j$ and for $k = 1,\cdots,K-1$, we compute*

$$\mathbf{y} = dM\mathbf{y} + \mathbf{x}_j.$$

Then writing $\mathbf{y} = [y_1,\cdots,y_N]^\top$, we let $\alpha = 1+y_j$ and $\epsilon = (P_j/\alpha)\mathbf{y}$. Thus, by (11), $\mathbf{P}_+ \approx \mathbf{P} - \epsilon$ within the given tolerance.

The above algorithm leads to sequentially updating $\mathbf{P}$, i.e. update one webpage or one keyword or one document at a time. See Remark 5.1 for a parallel update scheme.

3 Sociological Analysis

Sociologists have studied the linkages, networks for a long time. Many concepts of centrality measures have been created. In [7], Bonscich proposed a measure of centrality $\mathbf{c}(\alpha,\beta)$ with normalization factor α and parameter β which is a function of unit vectors of $\mathbb{R}^N$. For each unit vector, say, $\mathbf{v}_i$,

$$c_i(\alpha,\beta) = \sum_j (\alpha + \beta c_j(\alpha,\beta))R_{ij} \tag{12}$$

where R_{ij} is an entry of matrix relation $\mathbf{R}$ which may not be symmetric and main diagonal elements are zero. In matrix notation

$$\mathbf{c}(\alpha, \beta) = \alpha(I - \beta\mathbf{R})^{-1}\mathbf{R}\mathbf{1}. \tag{13}$$

See page 1173 of [7]. If we use the adjacent matrix for $\mathbf{R}$ and normalization factor $\alpha = (1-d)/N$, we see that (12) is just (2) when $\beta = d$, the damping factor. If $\alpha = (1-d)/N$ and $\mathbf{R}\mathbf{1} = \mathbf{1}$, we can see that (13) is the same as the equation on the right-hand side of (8). Using these two parameters, the PageRank is just $\mathbf{c}(\alpha, \beta)$. According to Bonacich, when $\mathbf{R}$ is asymmetric, e.g., $\mathbf{R} = M$, $\mathbf{c}(\alpha, \beta)$ measures prestige if $\beta > 0$. This gives a sociological justification that the Google's PageRank does measure the important relevance scores for each webpage.

Note that when $\mathbf{R}$ is symmetric, $\mathbf{c}(\alpha, \beta)$ measures centrality. Also, as in [7], β can be negative. For example, in a communication network,

As pointed out by Bonacich, his measure $\mathbf{c}(\alpha, \beta)$ seems hopelessly ambiguous,,

4 Improved Search Algorithms

We now explain the improvements discussed in the previous section in mathematical terminology. Recall all the notations and definitions in a previous section. We use page to denote a webpage, a keyword or phrase, a document, etc..

4.1 Convergence Analysis of In- and Out-Linkage Search Algorithm

For each page $\mathbf{v}$, $A_{\mathbf{v}}$ is the collection of all pages connected from $\mathbf{v}$. We now search all $\mathbf{u} \in A_{\mathbf{v}}$ to see if $\mathbf{u}$ connects to $\mathbf{v}$ also. We denote by $A_{\mathbf{v}}^+$ the collection of such page $\mathbf{u} \in A_{\mathbf{v}}$. Note that $A_{\mathbf{v}}^+$ could be an empty set if $\mathbf{v}$ is an unknown page to the world. Anyway, $A_{\mathbf{v}}^+ \subset A_{\mathbf{v}}$ is a subset. Let

$$\widetilde{L}(\mathbf{v}) = \#(A_{\mathbf{v}}) + \#(A_{\mathbf{v}}^+), \tag{14}$$

where $\#(A)$ stands for the number of entries in set A. It is called the cardinality of A. We define the new adjacency entry

$$\widetilde{\ell}(\mathbf{v}, \mathbf{u}) = \begin{cases} 1/\widetilde{L}(\mathbf{v}), & \text{if } \mathbf{u} \text{ is not connected to } \mathbf{v} \text{ from } \mathbf{u} \\ 2/\widetilde{L}(\mathbf{v}), & \text{if } \mathbf{u} \text{ is connected to } \mathbf{v} \\ 0 & \text{otherwise.} \end{cases} \tag{15}$$

According to our Plan 1, the new PageRank function $\widetilde{P}(\mathbf{v})$ is now defined by

$$\widetilde{P}(\mathbf{v}) = d \sum_{\mathbf{u} \in A_{\mathbf{v}}} \widetilde{P}(\mathbf{u})\widetilde{\ell}(\mathbf{u}, \mathbf{v}) + \frac{1-d}{N}, \quad \forall \text{ any unit vector } \mathbf{v} \in \mathbb{R}^N, \tag{16}$$

where $d \in (0, 1)$ is a damping factor. To find the value of $\widetilde{P}(\mathbf{v})$, we use similar iterative algorithm as in Section 2. Let

$$\widetilde{M} = \left[\widetilde{\ell}(\mathbf{v}_i, \mathbf{v}_j)\right]_{1 \le i,j \le N}, \tag{17}$$

We are now ready to define

Algorithm 4.1 (In- and Out-linkage Search Algorithm) *Starting from* $\widetilde{\mathbf{P}}_0 = \mathbf{1}/N$*, we iteratively compute*

$$\widetilde{P}_k = \frac{1-d}{N}\mathbf{1} + d\widetilde{M}\widetilde{P}_{k-1}, \quad \forall k = 1, 2, , ... \tag{18}$$

For convenience, we write $\widetilde{\mathbf{P}}$ to the new PageRank vector of all pages $\mathbf{v}_i, \in \mathbb{R}^N, i = 1, \cdots, N$.

Based on the analysis in Section 2, we can conclude the following

Theorem 4.1 *Let* $\widetilde{\mathbf{P}}$ *be the PageRank vector defined above. For all* $k \geq 1$*,*

$$\|\widetilde{\mathbf{P}}_k - \widetilde{\mathbf{P}}\|_1 \leq Cd^k, \quad \forall k \geq 1,$$

where C *is a positive constant independent of* k*.*

4.2 Convergence Analysis of Second Order linkage Search Algorithm

Next we study our improved search algorithm 2. In addition to use $A^+_{\mathbf{v}}$, we count the links from $\mathbf{u} \in A_{\mathbf{v}}$. That is, we look at $A_{\mathbf{u}}$. More number of links in $A_{\mathbf{u}}$, the less contribution to $\mathbf{u}$ according to the sociological justitication in the previous section. Thus, we define

$$n(\mathbf{v}) = \#(A_{\mathbf{v}}) + \sum_{\mathbf{u}\in A_{\mathbf{v}}} \frac{1}{\#(A_{\mathbf{u}})} \tag{19}$$

and

$$\widehat{\ell}(\mathbf{u}, \mathbf{v}) = \begin{cases} \dfrac{1 + \frac{1}{\#(A_{\mathbf{u}})}}{n(\mathbf{v})}, & \text{if } \mathbf{u} \in A_{\mathbf{v}} \\ 0 & \text{otherwise.} \end{cases} \tag{20}$$

It is easy to see

Lemma 4.1 *For any page* $\mathbf{v}$

$$\sum_{j=1}^{N} \widehat{\ell}(\mathbf{v}_j, \mathbf{v}) = 1. \tag{21}$$

Thus the new PageRank function $\widehat{P}(\mathbf{v})$ is now defined by

$$\widehat{P}(\mathbf{v}) = d \sum_{\mathbf{u}\in A_{\mathbf{v}}} \widehat{P}(\mathbf{u})\widehat{\ell}(\mathbf{u}, \mathbf{v}) + \frac{1-d}{N}, \quad \forall \text{ any unit vector } \mathbf{v} \in \mathbb{R}^N, \tag{22}$$

where $d \in (0, 1)$ is a damping factor. To find the value of $\widehat{P}(\mathbf{v})$, we do the iterations as before. Letting

$$\widehat{M} = \left[\widehat{\ell}(\mathbf{v}_i, \mathbf{v}_j)\right]_{1\leq i,j\leq N}, \tag{23}$$

Algorithm 4.2 (Second Order linkage Search Algorithm) *Starting from* $\widehat{\mathbf{P}}_0 = \mathbf{1}/N$*, we iteratively compute*

$$\widehat{P}_k = \frac{1-d}{N}\mathbf{1} + d\widehat{M}\widehat{P}_{k-1}, \quad \forall k = 1, 2, , ... \tag{24}$$

Similarly, we write $\widehat{\mathbf{P}}$ to the new PageRank vector of all pages $\mathbf{v}_i, \in \mathbb{R}^N, i = 1, \cdots, N$.

Based on the analysis in Section 2, we can conclude the following

Theorem 4.2 *Let $\widehat{\mathbf{P}}$ be the PageRank vector defined above. For all $k \geq 1$,*

$$\|\widehat{\mathbf{P}}_k - \widehat{\mathbf{P}}\|_1 \leq Cd^k, \quad \forall k \geq 1,$$

where C is a positive constant independent of k.

5 Yahoo Search Engine

Yahoo search engine is quite different from the Google search engine conceptually and hence, computationally. The following description of the Yahoo search engine is based on [38].

Yahoo emphasizes on the similarity of any two webpages, two documents, or two web hubs. There are many sources of similarity information including link similarity, text similarity, multimedia component similarity, click through similarity, title similarity, URL similarity, cache log similarity, and etc.. For simplicity, let us consider the search engine for documents. Even for documents, there are many categories of similarity such as title similarity, author similarity, text similarity and etc.. For example, the text similarity is the dot product of two vectors which are the frequency usages of various words. That is, suppose that two documents use total M words. For one document, let $\mathbf{x}$ be the vector of length M whose each entry is the frequency usage of a word in the document. Similar for vector $\mathbf{y}$ for the other document. Then $\mathbf{x} \cdot \mathbf{y}$ denote the similarity between these two documents.

Suppose that the base set of all on-line documents contains N documents. The global similarity matrix $W = [w_{ij}]_{1 \leq i,j \leq N}$ with $w_{ij} \geq 0$ representing the strength of the similarity between document i and document j. Assume that $w_{ij} \leq M$ for all $i, j = 1, 2, \cdots, N$. Suppose that there are K categories of various similarities. For each $k = 1, \cdots, K$, let $\mathbf{x}(k) = [x_{1k}, x_{2k}, \cdots, x_{Nk}] \in \mathbb{R}^N$ be a vector with nonnegative entries satisfying

$$\sum_{i=1}^{N} x_{ik} = 1.$$

It is a vector of scores or confidence values for all documents with respect to the k^{th} category of similarity. Note that the global similarity matrix combines all sources of similarity information together. Thus, $w_{ij} = f(w_{ij}(1), \cdots, w_{ij}(K))$ with $w_{ij}(k), k = 1, \cdots, K$ being the similarity between document i and document j with respect to the category k. Yahoo considers a global similarity objective function

$$P(\mathbf{x}) = \sum_{k=1}^{K} \mathbf{x}(k)^\top W \mathbf{x}(k) = \sum_{k=1}^{} \sum_{i,]=1}^{N} x_{ik} w_{ij} x_{jk}.$$

Yahoo engine will have to first do data preparation or preprocessing by generating the global similarity matrices W based on training and learning and then compute the matrix $\mathbf{x}$ to maximize the objective function value by solving the following maximization problem:

$$\max_{\substack{\mathbf{x}(k) \in C \\ 1 \leq k \leq K}} P(\mathbf{x}), \tag{25}$$

where $\mathbf{x}(k) = [x_{1k}, x_{2k}, \cdots, x_{Nk}]^\top$ and $C = \{(x_1, x_2, \cdots, x_n), x_i \geq 0, i = 1, \cdots, N, \sum_{i=1}^N x_i = 1\}$ is a convex set in $\mathbb{R}^N$. This solution matrix $\mathbf{x}$ determines the search results according to the numerical values in the entries of $\mathbf{x}$. For document i, the document j is most similar to document i if the distance between vectors $x_{ik}, k = 1, \cdots, K$ and $x_{jk}, k = 1, \cdots, K$ is the smallest.

Since $P(\mathbf{x})$ is a quadratic function of $\mathbf{x}$ and hence continuous, $P(\mathbf{x})$ achieves its maximum within the bounded set C. It is easy to see

Lemma 5.1 *Let $M = \max_{1 \leq i,j \leq n} w_{ij}$ be the maximum of the entries of the global similarity matrix. Then $P(\mathbf{x}) \leq MK$ for all $\mathbf{x}(k) \in C, k = 1, \cdots, K$.*

Proof. Since $\sum_{i=1}^n x_i = 1$ and $x_i \geq 0$ for all $i = 1, \cdots, N$, we have

$$P(\mathbf{x}) = \sum_{k=1}^K \sum_{ij} x_{ik} w_{ij}(k) x_{jk} \leq \sum_{k=1}^K \sum_{ij} M x_i x_j = \sum_{k=1}^K M \sum_{i=1}^n \sum_{j=1}^n x_{jk} x_{ik} = MK.$$

■

However, $P(\mathbf{x})$ may achieve its maximum at several vectors. For example, letting $K = 1$, $N = 3$ and W be the identity matrix of size 3×3, we know the maximum is 1 when $\mathbf{x} = (1, 0, 0), (0, 1, 0), (0, 0, 1)$. Certainly, this is just a pathological example as no one uses an identity matrix for the global similarity matrix. The algorithm Yahoo uses produces a unique maximizer each time as it will be discussed below.

To solve the maximization problem, Yahoo uses the Growth Transformation algorithmI—(GTA). The most important advantages of this algorithm are that it guarantees a monotonic increase sequence of the objective function values and the computation is stable as each step is a convex combination of the previous step. For simplicity, we assume that $K = 1$. In fact, Yahoo engine does a loop to go through each category using the following GTA.

Algorithm 5.1 (the Growth Transformation) *Starting with a trained vector $\mathbf{x}^{(1)}$ in C, for $k = 1, 2, \cdots$, one computes*

$$\mathbf{x}_j^{(k+1)} = \frac{\mathbf{x}_j^{(k)} \frac{\partial}{\partial x_j} P(\mathbf{x}^{(k)})}{\sum_{i=1}^N \mathbf{x}_i^{(k)} \frac{\partial}{\partial x_i} P(\mathbf{x}^{(k)})} \tag{26}$$

for $j = 1, \cdots, N$.

This algorithm is mainly based on an inequality proved in [5]. For convenience, we present a short version of the proof due to the degree $d = 2$.

Lemma 5.2 (Baum and Eagon, 1967) *Let $P(\mathbf{x})$ be a homogeneous polynomial of degree d with nonnegative coefficients. Let $\mathbf{x} \in C$ the convex defined above. For any $\mathbf{x} \in C$, let*

$$\mathbf{y} = (y_1, \cdots, y_N) \text{ with } y_j = \frac{x_j \frac{\partial}{\partial x_j} P(\mathbf{x})}{\sum_{i=1}^N x_i \frac{\partial}{\partial x_i} P(\mathbf{x})}, j = 1, \cdots, N.$$

Then $P(\mathbf{y}) > P(\mathbf{x})$ unless $\mathbf{y} = \mathbf{x}$.

Proof. For convenience, let us use $P(\mathbf{x}) = \mathbf{x}^\top W\mathbf{x}$ to prove this inequality, i.e., we assume that $d = 2$. Clearly, P is a homogeneous polynomial of degree $d = 2$. Since the entries of W are nonnegative, $P(\mathbf{x})$ satisfies the assumptions of this lemma. Note that

$$\frac{\partial}{\partial x_i}P(\mathbf{x}) = 2\sum_{j=1}^{n} w_{ij}x_j$$

and

$$\sum_{i=1}^{N} x_i \frac{\partial}{\partial x_i}P(\mathbf{x}) = 2P(\mathbf{x}).$$

Thus, we first use Cauchy-Schwarz's inequality and then the geometric and arithmetic mean inequality to have

$$\begin{aligned}
P(\mathbf{x}) = \sum_{i,j=1,\cdots,n} w_{ij}x_ix_j &= \sum_{i,j=1}^{n} (w_{ij}y_iy_j)^{1/3}\left(w_{ij}^{2/3}x_ix_j\frac{1}{(y_iy_j)^{1/3}}\right) \\
&\leq \left(\sum_{i,j=1}^{n} w_{ij}y_iy_j\right)^{1/3}\left(\sum_{i,j=1}^{n} w_{ij}x_ix_j\left(\frac{x_ix_j}{y_iy_j}\right)^{1/2}\right)^{2/3} \\
&\leq \left(\sum_{i,j=1}^{n} w_{ij}y_iy_j\right)^{1/3}\left(\sum_{i,j=1}^{n} w_{ij}x_ix_j\frac{1}{2}\left(\frac{x_i}{y_i}+\frac{x_j}{y_j}\right)\right)^{2/3}
\end{aligned}$$

Since $y_i = x_i\sum_{j=1}^{n} w_{ij}x_j/O(x)$, we have

$$\begin{aligned}
\sum_{i,j=1}^{n} w_{ij}x_ix_j\frac{1}{2}\left(\frac{x_i}{y_i}+\frac{x_j}{y_j}\right) &= \frac{1}{2}\sum_{i,j=1}^{n} w_{ij}x_ix_j\left(\frac{P(\mathbf{x})}{\sum_{j=1}^{n} w_{ij}x_j}+\frac{P(\mathbf{x})}{\sum_{i=1}^{n} w_{ij}x_i}\right) \\
&= \frac{P(\mathbf{x})}{2}\left(\sum_{i=1}^{n} x_i\sum_{j=1}^{n} w_{ij}x_j/\left(\sum_{k=1}^{n} w_{ik}x_k\right)+\sum_{j=1}^{n} x_j\sum_{i=1}^{n} w_{ij}x_i/\left(\sum_{k=1}^{n} w_{kj}x_k\right)\right) \\
&= \frac{P(\mathbf{x})}{2}\left(\sum_{i=1}^{n} x_i+\sum_{j=1}^{n} x_j\right) = P(\mathbf{x}).
\end{aligned}$$

since $\mathbf{x} \in C$. Combining the above discussion together, we have

$$P(\mathbf{x}) \leq \left(\sum_{i,j=1}^{n} w_{ij}y_iy_j\right)^{1/3}(P(\mathbf{x}))^{2/3}.$$

It thus follows that $P(\mathbf{x}) \leq \sum_{i,j=1}^{n} w_{ij}y_iy_j = P(\mathbf{y})$. Note that from the above discussion, the inequalities become equalities for Cauchy-Schwarz's inequality and geometric and arithmetic mean inequality when $\mathbf{y} = \mathbf{x}$. Therefore, we have an strictly inequality when $\mathbf{y} \neq \mathbf{x}$. We have thus completed the proof. ■

With this lemma, we are able to show the convergence of Algorithm 5.1. Due to the monotonically increasing property of $P(\mathbf{x}^{(k)})$ and the boundedness of $P(\mathbf{x})$ by Lemma 5.1, we see that $P(\mathbf{x}^{(k)})$ is convergent to $P(\mathbf{x}^*)$ for some point $\mathbf{x}^*$. In practice, the iterations may end in a finite steps. Note that for $\mathbf{x} \in C$, $\nabla P(\mathbf{x}) \neq 0$ and each component of $\nabla P(\mathbf{x})$ is strictly positive. Thus, objective function values are increasing until P achieves its maximum. In this case, by Lemma 5.2, we have $\mathbf{x}^{(n+1)} = \mathbf{x}^{(n)}$ for some integer n.

Also, $\mathbf{x}^{(k)}, k \geq 1$ are all in C and hence, are bounded. It follows that there exists a convergent subsequence. Due to the nonuniqueness of the maximizers, we can not conclude the convergence of the whole sequence.

Suppose that W is a positive definite, $P(\mathbf{x})$ is a convex function. when $\mathbf{x}$ and $\mathbf{y}$ defined in Lemma 5.2 are close enough, we have

$$\nabla P(\mathbf{x}) \cdot (\mathbf{y} - \mathbf{x}) \geq 0 \tag{27}$$

since $P(\mathbf{y}) > P(\mathbf{x})$. It follows that

$$P(\mathbf{y}) - P(\mathbf{x}) = P(\mathbf{y} - \mathbf{x}) + 2(\mathbf{y} - \mathbf{x})^\top W\mathbf{x} = P(\mathbf{y} - \mathbf{x}) + (\mathbf{y} - \mathbf{x})^\top \nabla P(\mathbf{x}) \geq \lambda_n \|\mathbf{y} - \mathbf{x}\|^2, \tag{28}$$

where λ_n denotes the smallest eigenvalue of W and $\|\mathbf{z}\|$ denotes the ℓ_2 norm of vector $\mathbf{z} \in \mathbb{R}^n$. Therefore we have

Theorem 5.1 *Suppose that W is symmetric and positive definite. Let M be the largest value of the entries of W. Let λ_n be the smallest eigenvalue of W. Let $\mathbf{x}^{(k)}, k \geq 1$ be the sequence from Algorithm 5.1. Then there exists a convergent subsequence, say $\mathbf{x}^{(m_k)}, k \geq 1$ which converge to a maximizer $\mathbf{x}^*$ and satisfy*

$$\sum_{k=1}^{\infty} \|\mathbf{x}^{(m_{k+1})} - \mathbf{x}^{(m_k)}\|^2 \leq C < \infty,$$

where C is a positive constant dependent on λ_n and M.

Proof. We have discussed that the sequence $\mathbf{x}^{(k)}$ has a convergent subsequence. For convenience, let us say $\mathbf{x}^{(k)}$ converges to $\mathbf{x}^*$. Thus, $\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}$ will be very close when $k \geq K_0$ for a positive inteeger K_0. Thus,

$$P(\mathbf{x}^{(k+1)}) - P(\mathbf{x}^{(k)}) \geq P(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) \geq \lambda_n \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2$$

by using (28) and (27). Hence,

$$\lambda_n \sum_{k \geq K_0} \|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\|^2 \leq M - P(\mathbf{x}^{(K_0)}) \leq M < \infty.$$

This completes the proof. ■

Although Yahoo does not discuss much mathematics behind their generating the global similarity matrix W, one approach called *matrix completion* can be used. The matrix completion is a research topic in mathematics which is recently actively studied. It starts from the well-known Netflix problem. Indeed, Netflix (cf. [36]) made available publicly such a set of data with about 10^5 known entries of its movie rating matrix of size about 5×10^5 times 2×10^5 and challenged the research community to complete the movie recommending matrix with root mean square error less than 0.8563. This matrix completion problem has been studied actively ever since. We refer the following papers, [26], [41], and [29] for in-depth study and the references therein.

6 Remarks

We have the following remarks in order:

Remark 6.1 *We will not discuss other link-based ranking systems for webpages such as the HITS algorithm invented by Jon Kleinberg (used by Ask.com), the IBM CLEVER project, IDD Information Services designed by Robin Li who found Baidu in China, the TrustRank algorithm and etc..*

Remark 6.2 *Next we discuss how to in parallel update the RageRank simultaneously for multiple webpages and keywords and names of documents at a time. We need the following*

Lemma 6.1 (Woodbury matrix identity) *Suppose that A is invertible and C is also invertible. If $A+UCV$ is invertible, then*

$$(A+UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1}+VA^{-1}U)^{-1}VA^{-1}.$$

The above formula can be rewritten in the following form

Lemma 6.2 *Suppose that U and V are unitary matrices and C is a diagonal matrix containing nonnegative values as entries. If $A+UCV$ is invertible, then*

$$(A+UCV^{\top})^{-1} = A^{-1} - (VC^{\dagger}U^{\top}A+I)^{-1}A^{-1}, \tag{29}$$

where $C^{\dagger}$ is the pseudo-inverse of C.

Proof. *First of all, we have*

$$(A+UCV^{\top})^{-1} = A^{-1} - A^{-1}U(C^{\dagger}+V^{\top}A^{-1}U)^{-1}V^{\top}A^{-1}$$

by using the same direct proof for Woodbury formula. Furthermore, we use the invertibility of U, V and A to rewrite the right-hand side of the above formula to get the equality in (29). ■

For the Google matrix $A = I - dM$ and $UCV^{\top} = -d\sum_{j=1}^{k}\mathbf{x}_j\mathbf{v}_j^{\top}$, we know most entries of matrix $\sum_{j=1}^{k}\mathbf{x}_j\mathbf{v}_j^{\top}$ of size $N \times N$ are zero. There is a nonzero block which is of size $n_k \times n_k$ with integer $n_k << N$ if k is not very big. For convenience, let us say the principal block of size $n_k \times n_k$ are nonzero. It is easy to find the singular value decomposition of this block, $-\sum_{j=1}^{k}\mathbf{x}_j\mathbf{v}_j^{\top} = U_1C_1V_1^{\top}$. Then let $U = diag(U_1, I_{N-n_k})$, $C = diag(C_1, \mathbf{0}_{N-n_k})$ and $V = diag(V_1, I_{N-n_k})$, where I_{N-n_k} is the identity matrix of size $N-n_k$ and $\mathbf{0}_{N-n_k}$ is zero block of size $N-n_k$. In general C_1 is not invertible. We can write it in the following form: if C_1 is of rank $r_k < n_k$,

$$C_1 = \begin{bmatrix} C_{11} & 0 \\ 0 & 0 \end{bmatrix},$$

where C_{11} is diagonal matrix containing all the nonzero singular values of $-\sum_{j=1}^{k}\mathbf{x}_j\mathbf{v}_j^\top$.

From (29), we need to find $(VC^\dagger U^\top A+I)^{-1}$. First, we have

$$VC^\dagger U^\top A=\begin{bmatrix}V_1 & 0\\ 0 & I_{N-n_k}\end{bmatrix}\begin{bmatrix}\frac{1}{d}C_{11}^{-1} & 0\\ 0 & \mathbf{0}_{N-r_k}\end{bmatrix}\begin{bmatrix}U_1^\top & 0\\ 0 & I_{N-n_k}\end{bmatrix}\begin{bmatrix}I_{n_k}-dM_{11} & -dM_{12}\\ -dM_{21} & I_{N-n_k}-dM_{22}\end{bmatrix}.$$

for a matrix $\widetilde{C}_1$ which is of size $n_k\times n_k$. Then we have

$$VC^\dagger U^\top A=\begin{bmatrix}\frac{1}{d}\widetilde{C}_1 & 0\\ 0 & \mathbf{0}_{N-n_k}\end{bmatrix}\begin{bmatrix}I_{n_k}-dM_{11} & -dM_{12}\\ -dM_{21} & I_{N-n_k}-dM_{22}\end{bmatrix},$$

where the Google matrix M is written in terms of the same division of blocks as $VC^\dagger U^\top$. Furthermore,

$$\begin{aligned}VC^\dagger U^\top A+I&=\begin{bmatrix}\frac{1}{d}\widetilde{C}_1(I_{n_k}-dM_{11})+I_{n_k} & -\widetilde{C}_1M_{12}\\ 0 & I_{N-n_k}\end{bmatrix}\\&=\begin{bmatrix}\frac{1}{d}\widetilde{C}_1 & 0\\ 0 & I_{N-n_k}\end{bmatrix}\left(\begin{bmatrix}d\widetilde{C}_1^{-1}-dM_{11} & -dM_{12}\\ 0 & \mathbf{0}_{N-n_k}\end{bmatrix}+I\right).\end{aligned}$$

Finally, we have

$$(VC^\dagger U^\top A+I)^{-1}=\left(I+d\begin{bmatrix}\widetilde{C}_1^\dagger-M_{11} & -M_{12}\\ 0 & \mathbf{0}_{N-n_k}\end{bmatrix}\right)^{-1}\begin{bmatrix}d\widetilde{C}_1^\dagger & 0\\ 0 & I_{N-n_k}\end{bmatrix}.$$

It follows the updated PageRank vector is

$$\begin{aligned}\mathbf{P}_+&=(I-dM-dUCV^\top)^{-1}\frac{1-d}{N}\mathbf{1}\\&=(I-dM)^{-1}\frac{1-d}{N}\mathbf{1}-\left(I+d\begin{bmatrix}\widetilde{C}_1^\dagger-M_{11} & -M_{12}\\ 0 & \mathbf{0}_{N-n_k}\end{bmatrix}\right)^{-1}\begin{bmatrix}d\widetilde{C}_1^\dagger & 0\\ 0 & I_{N-n_k}\end{bmatrix}(I-dM)^{-1}\frac{1-d}{N}\mathbf{1}\\&=\mathbf{P}-\left(I+d\begin{bmatrix}\widetilde{C}_1^\dagger-M_{11} & -M_{12}\\ 0 & \mathbf{0}_{N-n_k}\end{bmatrix}\right)^{-1}\begin{bmatrix}d\widetilde{C}_1^\dagger & 0\\ 0 & I_{N-n_k}\end{bmatrix}\mathbf{P}.\end{aligned}$$

We are now ready to present an algorithm for multiple updates.

Algorithm 6.1 (Multiple Updates) *Choose an integer K large enough such that d^K is within a given tolerance. We first compute U_1, C_1, V_1 as explained above. Then compute $\widetilde{C}_1$. For the current PageRank vector $\mathbf{P}$, modify it to be*

$$\widetilde{\mathbf{P}}=\begin{bmatrix}d\widetilde{C}_1^\dagger & 0\\ 0 & I_{N-n_k}\end{bmatrix}\mathbf{P}$$

and then compute the following iterations

$$\mathbf{P}_+\approx\mathbf{P}-\sum_{j=0}^{K}d^j\begin{bmatrix}\widetilde{C}_1^\dagger-M_{11} & -M_{12}\\ 0 & \mathbf{0}_{N-n_k}\end{bmatrix}^j\widetilde{\mathbf{P}}.$$

Similarly we can discuss how to compute M_+ efficiently when adding a new webpage/document/link to the database. We omit the detail here.

Remark 6.3 *The importance of webpages related to each keyword, document or webpage can also be calculated based on the number of hits from the world. However, this is very easily be scrolled up by some artifical hits.*

Aknowledgment: The authors would like to thank Dr. John Lavery for his encouragement and support which enable them to carry out the research in this paper.

References

[1] T. Agryzkov, J. L. Oliver, L. Tortosa, J. F. Vicent, An algorithm for ranking the nodes of an urban network based on the concept of PageRank vector. Appl. Math. Comput. 219 (2012), no. 4, 2186-2193.

[2] F. K. Andersson and S. D. Silvester, The Mathematics of Internet Search Engines, Acta Appl. Math. 104 (2008), 211–242.

[3] K. Avrachenkov, N. Litvak, D. Nemirovsky, N. Osipova, Monte Carlo methods in PageRank computation: when one iteration is sufficient, SIAM J. Numer. Anal. 45 (2007) 890-904.

[4] F. Backaker, The Google markov chain: convergence speed and eigenvalues, Master Thesis, Uppsala University, Sweden.

[5] L. Baum and R. C. Eagon, An inequality with applications to statistical estimation for probabilitic functions of Markov processes and to a model for ecology, Bull. Amer. Math. Soc., 73(1967), pp. 360–363.

[6] M. Benzi, E. Estrada, and C. Klymko, Ranking hubs and authorities using matrix functions, Linear Algebra and its Applications, 438(2013), 2447–2474.

[7] P. Bonachich, Power and centrality: a family of measures, Amer. J. Sociology, 92(1987), 1170–1182.

[8] C. Brezinski, M. Redivo-Zaglia, and S. Serra-Capizzano, Extrapolation methods for PageRank computations, C. R. Acad. Sci. Paris, Ser I. 340 (2005) 393–397.

[9] C. Brezinski and M. Redivo-Zaglia, The PageRank vector: properties, computation, approximation and acceleration, SIAM J. Matrix Anal. Appl. 28(2006), 551–575.

[10] Brin, S. and Page, L. (1998). The anatomy of a large-scale hypertextual Web search engine. Computer Networks and ISDN Systems 30: 107-117.

[11] M. Brinkmeier, PageRank Revisited, ACM Transactions on Internet Technology, 6(2006), 282-301.

[12] F. Chung and W. Zhao, A sharp PageRank algorithm with applications to edge ranking and graph sparsification. Algorithms and models for the web graph, 214, Lecture Notes in Comput. Sci., 6516, Springer, Berlin, 2010.

[13] F. Chung, A. Tsiatas, Finding and visualizing graph clusters using PageRank optimization, Internet Math. 8 (2012), no. 1-2, 46-72.

[14] L. Eden, A Note on the Eigenvalues of the Google Matrix, arXiv:math/0401177v1, 2004.

[15] D. Gleich, L. Zhukov and P. Berkhin, Fast Parallel PageRank: A Linear System Approach, Technical Report L-2004-038, Yahoo! Research Labs, 2004.

[16] G. H. Golub and C. Greif, An Arnold type algorithm for computing PageRank, BIT Numerical Mathematics, 46(2006), 759–771.

[17] G. H. Golub and C. D. Meyer, Using the QR factorization and group inverse to compute, differentiate, and estimate the sensitivity of stationary probabilities for Markov chains, SIAM J. Algebra Discr., 17 (1986), pp. 273-281.

[18] I. C. F. Ipsen, R. S. Wills, Mathematical properties and analysis of Google's PageRank. Bol. Soc. Esp. Mat. Apl. Se MA No. 34 (2006), 191-196.

[19] [I.C.F. Ipsen, T.M. Selee, PageRank computation, with special attention to dangling nodes, SIAM J. Matrix Anal. Appl. 29 (2007) 1281–1296.

[20] H. Ishii, R. Tempo, E.-W. Bai, Distributed randomized PageRank algorithms based on web aggregation over unreliable channels, in: Proc. 49th IEEE Conf. on Decision and Control, pp. 6602-6607, 2010.

[21] H. Ishii, R. Tempo, E.-W. Bai, Distributed randomized PageRank algorithms over unreliable channels, in: L. Qiu, et al. (Eds.), Developments in Control Theory towards Glocal Control, Institution of Engineering and Technology, 2012, pp. 147-156.

[22] H. Ishii, R. Tempo, and E. W. Bai, PageRank computation via a distributed randomized approach with lossy communication, Systems & Control Letters 61 (2012) 1221-1228.

[23] http://en.wikipedia.org/wiki/PageRank

[24] Haveliwala, T. and S. Kamvar. (2003). The Second Eigenvalue of the Google Matrix. Stanford University Technical Report: 7056. arXiv:math/0307056.

[25] S. Kamvar, T. Haveliwala, G. Golub, Adaptive methods for the computation of PageRank, Linear Algebra Appl. 386 (2004) 51-65.

[26] R.H. Keshavan, A. Montanari, and S. Oh. Matrix completion from a few entries. IEEE Transactions on Information Theory, 56(6):2980–2998, 2010.

[27] Kincaid, D. and W. Cheney. (2002). Numerical Analysis: Mathematics of Scientific Computing, Brooks/Cole.

[28] J. Kleinberg, Authoritative sources in a hyperlinked environment, J. ACM 46 (1999) 604-632.

[29] M. J. Lai, Yin, W. T. and Xu, Y. Y., On unconstrained nonconvex minimizations for sparse vector and low-rank matrix recovery, accepted for publication in SIAM J. Numer. Analysis, (2013).

[30] A. N. Langville and C. D. Meyer, Deeper inside PageRank, Internet Math., 1 (2005), pp. 335-380.

[31] A. N. Langville and C. D. Meyer, A survey of eigenvector methods for Web information retrieval, SIAM Rev., 47 (2005), pp. 135-161.

[32] A. N. Langville and C. D. Meyer, Updating Markov chains with an eye on Google's PageRank, SIAM J. Matrix Anal. Appl. 27(2006), 968–987.

[33] A. Longville and C. Meyer, Google's PageRank and Beyond: The Science of Search Engine Rankings, Princeton Univ. Press, 2006.

[34] Y. Lin, X. Shi, Y. Wei, On computing PageRank via lumping the Google matrix, Journal of Computational and Applied Mathematics 224 (2009) 702–708.

[35] R. Nath and N. Kumar, To Overcome HITS Rank Similarity Confliction of Web Pages using Weight Calculation and Rank Improvement, Amer. Inst. Physics Conf. Proc. 1414 (2011), 77 doi: 10.1063/1.3669935.

[36] Netflix Prize at http://www.netflixprize.com/.

[37] Page, L., (1997). "PageRank: Bringing Order to the Web", Stanford Digital Library Project, talk. August 18, 1997 (archived 2002).

[38] M. Palmer, G. Sun, and H. Zha, United States Patent No. 6,990,628 B1, Filed on June 14, 1999 and Date of Patent Jan. 24, 2006.

[39] S. Serra-Capizzano, Jordan canonical form of the Google matrix: a potential contribution to the PageRank computation, SIAM J. Matrix Anal. Appl., 27(2005), 305–312.

[40] J. Wu and Y. Tan, PageRank algorithm optimization and improvement, Computer Engineering and Application, 45(2009), 56–59.

[41] Z. Wen, W. Yin, and Y. Zhang. Solving a low-rank factorization model for matrix completion by a nonlinear successive over-relaxation algorithm. Rice University CAAM Report TR10-07, 2010.